ENSF 594 Lab Assignment 3

Zach Frena

Complexity Analysis:

What is the worst-case complexity of your algorithm when checking if two words are anagrams of each other? Express this using big-O notation and use the variable k to represent the number of letters in each word.

Assuming that:

k = number of letters in each word

n = number of words in the input file

To check whether 2 words are anagrams of each other, I used a 2-step method:

1) **Alphabetic sort process**:
- Systematically loops through each word in the input file and sorts the word alphabetically using an insertion sort algorithm. This sorting algorithm has a complexity of $O(k^2)$.

2) **Comparison process:**

- Traverse through the list of (now) alphabetically sorted words and perform sequential comparison. The act of comparing every word to every other word requires a nested for-loop, and therefore has a complexity of $O(n^2)$.

Using the above information, this means that the complexity required to check if any two (or more) words are anagrams of each other is $O(k^2) + O(n^2) = O(k^2 + n^2)$.

Citations:

"Insertion Sort," *GeeksforGeeks*, 08-Jul-2021. [Online]. Available: https://www.geeksforgeeks.org/insertion-sort/. [Accessed: 20-Jul-2021].